How does Lipschitz Regularization Influence GAN Training?

Yipeng Qin[1,3], Niloy Mitra[2], and Peter Wonka[3]

[1] Cardiff University, qiny16@cardiff.ac.uk
[2] UCL/Adobe Research, n.mitra@cs.ucl.ac.uk
[3] KAUST, pwonka@gmail.com

Abstract. Despite the success of Lipschitz regularization in stabilizing GAN training, the exact reason of its effectiveness remains poorly understood. The direct effect of K-Lipschitz regularization is to restrict the $L2$-norm of the neural network gradient to be smaller than a threshold K (e.g., $K = 1$) such that $\|\nabla f\| \leq K$. In this work, we uncover an even more important effect of Lipschitz regularization by examining its impact on the loss function: *It degenerates GAN loss functions to almost linear ones by restricting their domain and interval of attainable gradient values.* Our analysis shows that loss functions are only successful if they are degenerated to almost linear ones. We also show that loss functions perform poorly if they are not degenerated and that a wide range of functions can be used as loss function as long as they are sufficiently degenerated by regularization. Basically, Lipschitz regularization ensures that all loss functions *effectively work in the same way.* Empirically, we verify our proposition on the MNIST, CIFAR10 and CelebA datasets.

Keywords: Generative adversarial network (GAN) · Lipschitz regularization · loss functions

1 Introduction

Generative Adversarial Networks (GANs) are a class of generative models successfully applied to various applications, e.g., pose-guided image generation [17], image-to-image translation [29, 23], texture synthesis [5], high resolution image synthesis [27], 3D model generation [28], urban modeling [13]. Goodfellow et al. [7] proved the convergence of GAN training by assuming that the generator is always updated according to the temporarily optimal discriminator at each training step. In practice, this assumption is too difficult to satisfy and GANs remain difficult to train. To stabilize the training of the GANs, various techniques have been proposed regarding the choices of architectures [24, 10], loss functions [2, 19], regularization and normalization [2, 8, 20, 21]. We refer interested readers to [16, 14] for extensive empirical studies.

Among them, the Lipschitz regularization [8, 21] has shown great success in stabilizing the training of various GANs. For example, [18] and [4] observed that the gradient penalty Lipschitz regularizer helps to improve the training of

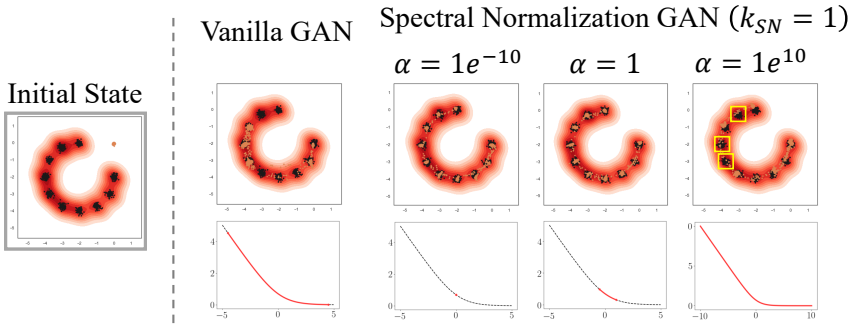
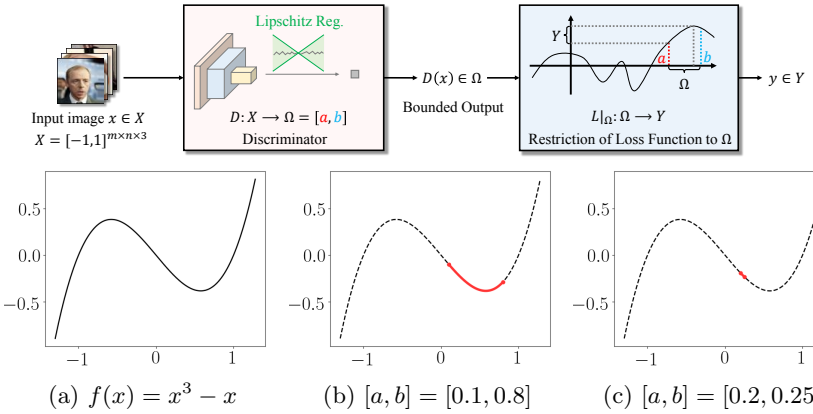


Fig. 1: An illustrative 2D example. First row: model distribution (orange point) vs. data distribution (black points). Second row: domains of the loss function (red curve). It can be observed that the performance of spectral normalized GANs [21] worsen when their domains are enlarged ($\alpha = 1e^{10}$, yellow boxes). However, good performance can always be achieved when their domains are restricted to near-linear ones ($\alpha = 1e^{-10}$ and $\alpha = 1$). Please see Sections 3.2 and 4.1 for the definitions of α and k_{SN}, respectively.

the LS-GAN [19] and the NS-GAN [7], respectively; [21] observed that the NS-GAN, with their spectral normalization Lipschitz regularizer, works better than the WGAN [2] regularized by gradient penalty (WGAN-GP) [8].

In this paper, we provide an analysis to better understand the *coupling* of Lipschitz regularization and the choice of loss function. Our main insight is that the rule of thumb of using small Lipschitz constants (e.g., $K = 1$) is degenerates the loss functions to almost linear ones by restricting their domain and interval of attainable gradient values (see Figure 1). These degenerate losses improve GAN training. Because of this, the exact shapes of the loss functions before degeneration do not seem to matter that much. We demonstrate this by two experiments. First, we show that when K is sufficiently small, even GANs trained with non-standard loss functions (e.g., cosine) give comparable results to all other loss functions. Second, we can directly degenerate loss functions by introducing domain scaling. This enables successful GAN training for a wide range of K for all loss functions, which only worked for the Wasserstein loss before. Our contributions include:

- We discovered an important effect of Lipschitz regularization. It restricts the domain of the loss function (Figure 2).
- Our analysis suggests that although the choice of loss functions matters, the successful ones currently being used are all near-linear within the chosen small domains and actually work in the same way.



(a) $f(x) = x^3 - x$ (b) $[a, b] = [0.1, 0.8]$ (c) $[a, b] = [0.2, 0.25]$

Fig. 2: First row: Applying Lipschitz regularization restricts the domain of the loss function to an interval $\Omega = [a, b]$. Second row: Illustration of the domain restriction. We take a third-order polynomial loss function $f(x) = x^3 - x$ as an example. Its restricted domain $[a, b]$ is shown in red. (a) without restriction $f(x)$ is non-convex. (b) Restricting the domain of $f(x)$ makes it convex. (c) $f(x)$ is almost linear when its domain is restricted to a very small interval.

2 Related Work

2.1 GAN Loss Functions

A variety of GAN loss functions have been proposed from the idea of understanding the GAN training as the minimization of statistical divergences. Goodfellow et al. [7] first proposed to minimize the Jensen-Shannon (JS) divergence between the model distribution and the target distribution. In their method, the neural network output of the discriminator is first passed through a sigmoid function to be scaled into a probability in $[0, 1]$. Then, the cross-entropy loss of the probability is measured. Following [4], we refer to such loss as the *minimax* (MM) loss since the GAN training is essentially a minimax game. However, because of the saturation at both ends of the sigmoid function, the MM loss can lead to vanishing gradients and thus fails to update the generator. To compensate for it, Goodfellow et al. [7] proposed a variant of MM loss named the *non-saturating* (NS) loss, which heuristically amplifies the gradients when updating the generator.

Observing that the JS divergence is a special case of the f-divergence, Nowozin et al. [22] extended the idea of Goodfellow et al. [7] and showed that any f-divergence can be used to train GANs. Their work suggested a new direction of improving the performance of the GANs by employing "better" divergence measures.

Following this direction, Arjovsky et al. first pointed out the flaws of the JS divergence used in GANs [1] and then proposed to use the Wasserstein distance

instead (WGAN) [2]. In their implementation, the raw neural network output of the discriminator is directly used (i.e. the WGAN loss function is an identity function) instead of being passed through the sigmoid cross-entropy loss function. However, to guarantee that their loss is a valid Wasserstein distance metric, the discriminator is required to be Lipschitz continuous. Such requirement is usually fulfilled by applying an extra Lipschitz regularizer to the discriminator. Meanwhile, Mao et al. [19] proposed the Least-Square GAN (LS-GAN) to minimize the Pearson χ^2 divergence between two distributions. In their implementation, the sigmoid cross-entropy loss is replaced by a quadratic loss.

2.2 Lipschitz Regularization

The first practice of applying the Lipschitz regularization to the discriminator came together with the WGAN [2]. While at that time, it was not employed to improve the GAN training but just a requirement of the Kantorovich-Rubinstein duality applied. In [2], the Lipschitz continuity of the discriminator is enforced by *weight clipping*. Its main idea is to clamp the weights of each neural network layer to a small fixed range $[-c, c]$, where c is a small positive constant. Although weight clipping guarantees the Lipschitz continuity of the discriminator, the choice of parameter c is difficult and prone to invalid gradients.

To this end, Gulrajani et al. [8] proposed the *gradient penalty* (GP) Lipschitz regularizer to stabilize the WGAN training, i.e. WGAN-GP. In their method, an extra regularization term of discriminator's gradient magnitude is weighted by parameter λ and added into the loss function. In [8], the gradient penalty regularizer is one-centered, aiming at enforcing 1-Lipschitz continuity. While Mescheder et al. [20] argued that the zero-centered one should be more reasonable because it makes the GAN training converge. However, one major problem of gradient penalty is that it is computed with finite samples, which makes it intractable to be applied to the entire output space. To sidestep this problem, the authors proposed to heuristically sample from the straight lines connecting model distribution and target distribution. However, this makes their approach heavily dependent on the support of the model distribution [21].

Addressing this issue, Miyato et al. [21] proposed the *spectral normalization* (SN) Lipschitz regularizer which enforces the Lipschitz continuity of a neural network in the operator space. Observing that the Lipschitz constant of the entire neural network is bounded by the product of those of its layers, they break down the problem to enforcing Lipschitz regularization on each neural network layer. These simplified sub-problems can then be solved by normalizing the weight matrix of each layer according to its largest singular value.

3 Restrictions of GAN Loss Functions

In this section, first we derive why a K-Lipschitz regularized discriminator *restricts* the domain and interval of attainable gradient values of the loss function to intervals bounded by K (Section 3.1). Second, we propose a scaling method to restrict the domain of the loss function without changing K (Section 3.2).

3.1 How does the Restriction Happen?

Let us consider a simple discriminator $D(x) = L(f(x))$, where x is the input, f is a neural network with scalar output, L is the loss function. During training, the loss function L works by backpropagating the gradient $\nabla L = \partial L(f(x))/\partial f(x)$ to update the neural network weights:

$$\frac{\partial D(x)}{\partial W^n} = \frac{\boldsymbol{\partial L(f(x))}}{\boldsymbol{\partial f(x)}} \frac{\partial f(x)}{\partial W^n} \tag{1}$$

where W^n is the weight matrix of the n-th layer. Let X and Ω be the domain and the range of f respectively (i.e., $f : X \to \Omega$), it can be easily derived that the attainable values of ∇L is determined by Ω (i.e., $\nabla L : \Omega \to \Psi$). Without loss of generality, we assume that $x \in X = [-1, 1]^{m \times n \times 3}$ are normalized images and derive the bound of the size of Ω as follows:

Theorem 1. *If the discriminator neural network f satisfies the k-Lipschitz continuity condition, we have $f : X \to \Omega \subset \mathbb{R}$ satisfying $|\min(\Omega) - \max(\Omega)| \leq k\sqrt{12mn}$.*

Proof. Given a k-Lipschitz continuous neural newtork f, for all $x_1, x_2 \in X$, we have:

$$|f(x_1) - f(x_2)| \leq k\|x_1 - x_2\|. \tag{2}$$

Let $x_b, x_w \in X$ be the pure black and pure white images that maximize the Euclidean distance:

$$\|x_b - x_w\| = \sqrt{(-1-1)^2 \cdot m \cdot n \cdot 3} = \sqrt{12mn}. \tag{3}$$

Thus, we have:

$$\begin{aligned} |f(x_1) - f(x_2)| &\leq k\|x_1 - x_2\| \\ &\leq k\|x_b - x_w\| = k\sqrt{12mn}. \end{aligned} \tag{4}$$

Thus, the range of f is restricted to Ω, which satisfies:

$$|\min(\Omega) - \max(\Omega)| \leq k\sqrt{12mn} \tag{5}$$

Theorem 1 shows that the size of Ω is bounded by k. However, k can be unbounded when Lipschitz regularization is not enforced during training, which results in an unbounded Ω and a large interval of attainable gradient values. On the contrary, when K-Lipschitz regularization is applied (i.e., $k \leq K$), the loss function L is restricted as follows:

Corollary 1 (Restriction of Loss Function). *Assume that f is a Lipschitz regularized neural network whose Lipschitz constant $k \leq K$, the loss function L is C^2-continuous with M as the maximum absolute value of its second derivatives in its domain. Let Ψ be the interval of attainable gradient values that $\nabla L : \Omega \to \Psi$, we have*

$$\big|\min(\Omega) - \max(\Omega)\big| \leq K\sqrt{12mn} \tag{6}$$

$$\big|\min(\Psi) - \max(\Psi)\big| \leq M \cdot K\sqrt{12mn} \tag{7}$$

Corollary 1 shows that under a mild condition (C^2-continuous), applying K-Lipschitz regularization restricts the domain Ω and thereby the interval of attainable gradient values Ψ of the loss function L to intervals bounded by K. When K is small, e.g., $K = 1$ [8, 4, 18, 21], the interval of attainable gradient values of the loss function is considerably reduced, which prevents the backpropagation of vanishing or exploding gradients and thereby stabilizes the training. Empirically, we will show that these restrictions are indeed significant in practice and strongly influence the training.

Change in Ω_i During Training. So far we analyzed the restriction of the loss function by a static discriminator. However, the discriminator neural network f is dynamically updated during training and thus its range $\Omega^{\cup} = \cup_i \Omega_i$, where Ω_i is the discriminator range at each training step i. Therefore, we need to analyze two questions:

(i) How does the size of Ω_i change during training?

(ii) Does Ω_i shift during training (Figure 3)?

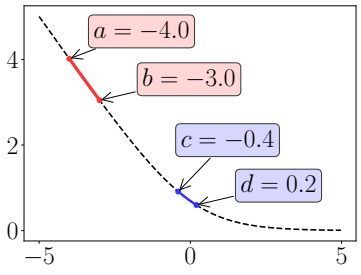


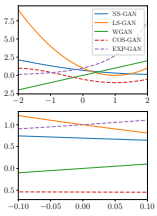
Fig. 3: Domain $[a, b]$ shifts to $[c, d]$.

For question (i), the size of Ω_i is always bounded by the Lipschitz constant K throughout the training (Corollary 1). For question (ii), the answer depends on the discriminator loss function:

- The shifting of Ω_i is prevented if the loss function is strictly convex. For example, the discriminator loss function of NS-GAN [7] (Table 1) is strictly convex and has a unique minimum when $f(x) = f(g(z)) = 0$ at convergence. Thus, minimizing it forces Ω_i to be positioned around 0 and prevents it from shifting. The discriminator loss function of LS-GAN [19] (Table 1) has a similar behavior. Its Ω_i is positioned around 0.5, since its minimum is achieved when $f(x) = f(g(z)) = 0.5$ at convergence. In this scenario, the Ω_i is relatively fixed throughout the training. Thus, $\Omega^{\cup}$ is still roughly bounded by the Lipschitz constant K.
- When the discriminator loss functions is not strictly convex, Ω_i may be allowed to shift. For example, the WGAN [2] discriminator loss function (Table 1) is linear and achieves its minimum when $f(x) = f(g(z))$ at convergence. Thus, it does not enforce the domain Ω_i to be fixed. However, the linear WGAN loss function has a constant gradient that is independent of Ω_i. Thus, regarding to the interval of attainable gradient values (Eq.7), we can view it as a degenerate loss function that still fits in our discussion. Interestingly, we empirically observed that the domain Ω_i of WGANs also get relatively fixed at late stages of the training (Figure 4).

Table 1: The GAN loss functions used in our experiments. $f(\cdot)$ is the output of the discriminator neural network; $g(\cdot)$ is the output of the generator; x is a sample from the training dataset; z is a sample from the noise distribution. LS-GAN$^{\#}$ is the zero-centered version of LS-GAN [18], and for the NS-GAN, $f^*(\cdot) = \text{sigmoid}[f(\cdot)]$. The figure on the right shows the shape of the loss functions at different scales. The dashed lines show non-standard loss functions: cos and exp.

GAN types	Discriminator Loss	Generator Loss
NS-GAN	$L_D = -\mathbb{E}[\log(f^*(x))] - \mathbb{E}[\log(1 - f^*(g(z)))]$	$L_G = -\mathbb{E}[\log(f^*(g(z)))]$
LS-GAN	$L_D = \mathbb{E}[(f(x) - 1)^2] + \mathbb{E}[f(g(z))^2]$	$L_G = \mathbb{E}[(f(g(z)) - 1)^2]$
LS-GAN$^{\#}$	$L_D = \mathbb{E}[(f(x) - 1)^2] + \mathbb{E}[(f(g(z)) + 1)^2]$	$L_G = \mathbb{E}[(f(g(z)) - 1)^2]$
WGAN	$L_D = \mathbb{E}[f(x)] - \mathbb{E}[f(g(z))]$	$L_G = \mathbb{E}[f(g(z))]$
COS-GAN	$L_D = -\mathbb{E}[\cos(f(x) - 1)] - \mathbb{E}[\cos(f(g(z)) + 1)]$	$L_G = -\mathbb{E}[\cos(f(g(z)) - 1)]$
EXP-GAN	$L_D = \mathbb{E}[\exp(f(x))] + \mathbb{E}[\exp(-f(g(z)))]$	$L_G = \mathbb{E}[\exp(f(g(z)))]$



3.2 Restricting Loss Functions by Domain Scaling

As discussed above, applying K-Lipschitz regularization not only restricts the gradients of the discriminator, but as a side effect also restricts the domain of the loss function to an interval Ω. However, we would like to investigate these two effects separately. To this end, we propose to decouple the restriction of Ω from the Lipschitz regularization by scaling the domain of loss function L by a positive constant α as follows,

$$L_\alpha(\Omega) = L(\alpha \cdot \Omega)/\alpha. \tag{8}$$

Note that the α in the denominator helps to preserve the gradient scale of the loss function. With this scaling method, we can effectively restrict L to an interval $\alpha \cdot \Omega$ without adjusting K.

Degenerate Loss Functions. To explain why this works, we observe that any loss function degenerates as its domain Ω shrinks to a single value. According to Taylor's expansion, let $\omega, \omega + \Delta\omega \in \Omega$, we have:

$$L(\omega + \Delta\omega) = L(\omega) + \frac{L'(\omega)}{1!}\Delta\omega + \frac{L''(\omega)}{2!}(\Delta\omega)^2 + \cdots. \tag{9}$$

As $|\max(\Omega) - \min(\Omega)|$ shrinks to zero, we have $L(\omega + \Delta\omega) \approx L(\omega) + L'(\omega)\Delta\omega$ showing that we can approximate any loss function by a linear function with constant gradient as its domain Ω shrinks to a single value. Let $\omega \in \Omega$, we implement the degeneration of a loss function by scaling its domain Ω with an extremely small constant α:

$$\lim_{\alpha \to 0} \frac{\partial L_\alpha(\omega)}{\partial \omega} = \frac{1}{\alpha} \cdot \frac{\partial L(\alpha \cdot \omega)}{\partial \omega} = \frac{\partial L(\alpha \cdot \omega)}{\partial(\alpha \cdot \omega)} = \nabla L(0). \tag{10}$$

In our work, we use $\alpha = 1e^{-25}$, smaller values are not used due to numerical errors (NaN).

4 Experiments

To support our proposition, first we empirically verify that applying K-Lipschitz regularization to the discriminator has the side-effect of restricting the domain and interval of attainable gradient values of the loss function. Second, with the proposed scaling method (Section 3.2), we investigate how the varying restrictions of loss functions influence the performance of GANs when the discriminator is regularized with a fixed Lipschitz constant. Third, we show that restricting the domain of any loss function (using decreasing α) converges to the same (or very similar) performance as WGAN-SN.

4.1 Experiment Setup

General Setup. In the following experiments, we use two variants of the standard CNN architecture [24, 2, 21] for the GANs to learn the distributions of the MNIST, CIFAR10 datasets at 32×32 resolution and the CelebA dataset [15] at 64×64 resolution. Details of the architectures are shown in the supplementary material. We use a batch size of 64 to train the GANs. Similar to [2], we observed that the training could be unstable with a momentum-based optimizer such as Adam, when the discriminator is regularized with a very small Lipschitz constant K. Thus, we choose to use an RMSProp optimizer with learning rate 0.00005. To make a fair comparison, we fix the number of discriminator updates in each iteration $n_{dis} = 1$ for all the GANs tested (i.e., we do not use multiple discriminator updates like [1, 2]). Unless specified, we stop the training after 10^5 iterations.

Lipschitz Regularizers. In general, there are two state-of-the-art Lipschitz regularizers: the gradient penalty (GP) [8] and the spectral normalization (SN) [21]. In their original settings, both techniques applied only 1-Lipschitz regularization to the discriminator. However, our experiments require altering the Lipschitz constant K of the discriminator. To this end, we propose to control K for both techniques by adding parameters k_{GP} and k_{SN}, respectively.

- For the gradient penalty, we control its strength by adjusting the target gradient norm k_{GP},

$$L = L_{GAN} + \lambda \mathop{\mathbb{E}}_{\hat{x} \in P_{\hat{x}}} [(\| \bigtriangledown_{\hat{x}} D(\hat{x})\| - k_{GP})^2], \tag{11}$$

  where L_{GAN} is the GAN loss function without gradient penalty, λ is the weight of the gradient penalty term, $P_{\hat{x}}$ is the distribution of linearly interpolated samples between the target distribution and the model distribution [8]. Similar to [8, 4], we use $\lambda = 10$.
- For the spectral normalization, we control its strength by adding a weight parameter k_{SN} to the normalization of each neural network layer,

$$\bar{W}_{SN}(W, k_{SN}) := k_{SN} \cdot W/\sigma(W), \tag{12}$$

  where W is the weight matrix of each layer, $\sigma(W)$ is its largest singular value.

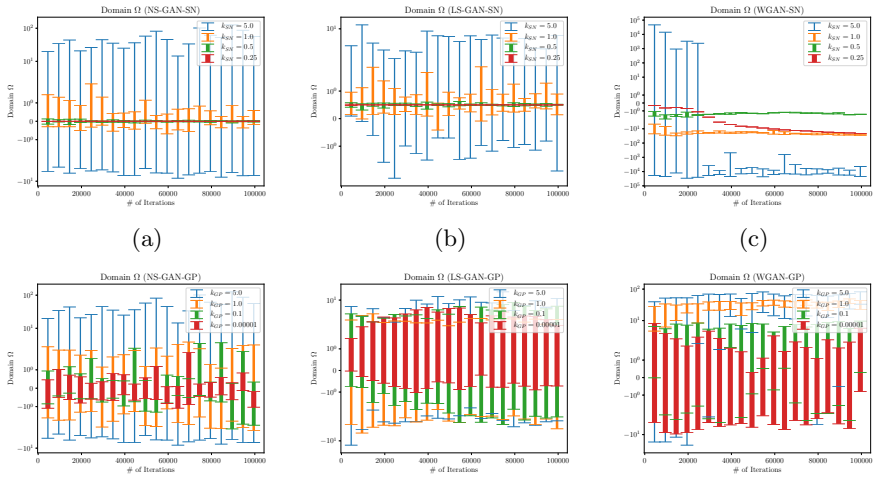


Fig. 4: Relationship between domain Ω and k_{GP}, k_{SN} for different loss functions on CelebA dataset, where k_{GP}, k_{SN} are the parameters controlling the strength of the Lipschitz regularizers. The domain Ω shrinks with decreasing k_{GP} or k_{SN}. Each column shares the same loss function while each row shares the same Lipschitz regularizer. NS-GAN: Non-Saturating GAN [7]; LS-GAN: Least-Square GAN [19]; WGAN: Wasserstein GAN [2]; GP: gradient penalty [8]; SN: spectral normalization [21]. Note that the y-axis is in log scale.

The relationship between k_{SN} and K can be quantitatively approximated as $K \approx k_{SN}^n$ [21], where n is the number of neural network layers in the discriminator. While for k_{GP}, we can only describe its relationship against K qualitatively as: the smaller k_{GP}, the smaller K. The challenge on finding a quantitative approximation resides in that the gradient penalty term $\lambda \mathbb{E}_{\hat{x} \in P_{\hat{x}}}[(\| \bigtriangledown_{\hat{x}} D(\hat{x})\| - k_{GP})^2]$ has no upper bound during training (Eq.11). We also verified our claims using Stable Rank Normalization (SRN)+SN [25] as the Lipschitz regularizer, whose results are shown in the supplementary material.

Loss Functions. In Table 1 we compare the three most widely-used GAN loss functions: the Non-Saturating (NS) loss function [7], the Least-Squares (LS) loss function [19] and the Wasserstein loss function [2]. In addition, we also test the performance of the GANs using some non-standard loss functions, $\cos(\cdot)$ and $\exp(\cdot)$, to support the observation that the restriction of the loss function is the dominating factor of Lipschitz regularization. Note that both the $\cos(\cdot)$ and $\exp(\cdot)$ loss functions are (locally) convex at convergence, which helps to prevent shifting Ω_i (Section 3.1).

Quantitative Metrics. To quantitatively measure the performance of the GANs, we follow the best practice and employ the Fréchet Inception Distance (FID)

Table 2: Domain Ω and the interval of attained gradient values $\nabla L(\Omega)$ against k_{SN} on the CelebA dataset.

k_{SN}	Ω	$\nabla L(\Omega)$
5.0	$[-8.130, 126.501]$	$[-1.000, -0.000]$
1.0	$[-0.683, 2.091]$	$[-0.664, -0.110]$
0.5	$[-0.178, 0.128]$	$[-0.545, -0.468]$
0.25	$[-0.006, 0.006]$	$[-0.502, -0.498]$

(a) NS-GAN-SN, $L(\cdot) = -\log(\text{sigmoid}(\cdot))$

k_{SN}	Ω	$\nabla L(\Omega)$
5.0	$[-2.460, 12.020]$	$[-4.921, 24.041]$
1.0	$[-0.414, 1.881]$	$[-0.828, 3.762]$
0.5	$[0.312, 0.621]$	$[0.623, 1.242]$
0.25	$[0.478, 0.522]$	$[0.956, 1.045]$

(b) LS-GAN-SN, $L(\cdot) = (\cdot)^2$

metric [11] in our experiments. The smaller the FID score, the better the performance of the GAN. The results on other metrics, *i.e.* Inception scores [3] and Neural Divergence [9], are shown in the supplementary material.

4.2 Empirical Analysis of Lipschitz Regularization

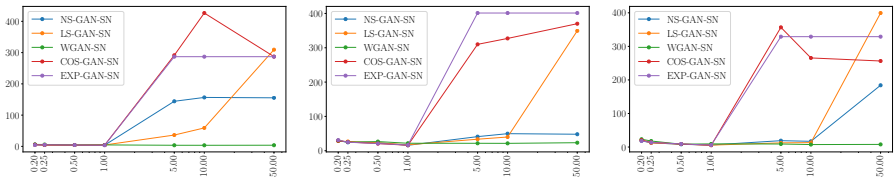
In this section, we empirically analyze how varying the strength of the Lipschitz regularization impacts the domain, interval of attained gradient values, and performance (FID scores) of different loss functions (Section 3.1).

Domain vs. Lipschitz Regularization. In this experiment, we show how the Lipschitz regularization influences the domain of the loss function. As Figure 4 shows, we plot the domain Ω as intervals for different iterations under different k_{GP} and k_{SN} for the gradient penalty and the spectral normalization regularizers respectively. It can be observed that: (i) For both regularizers, the interval Ω shrinks as k_{GP} and k_{SN} decreases. However, k_{SN} is much more impactful than k_{GP} in restricting Ω. Thus, we use spectral normalization to alter the strength of the Lipschitz regularization in the following experiments. (ii) For NS-GANs and LS-GANs, the domains Ω_i are rather fixed during training. For WGANs, the domains Ω_i typically shift at the beginning of the training, but then get relatively fixed in later stages.

Interval of Attained Gradient Values vs. Lipschitz Regularization. Similar to the domain, the interval of attained gradient values of the loss function also shrinks with the increasing strength of Lipschitz regularization. Table 2 shows the corresponding interval of attained gradient values of the NS-GAN-SN and LS-GAN-SN experiments in Figure 4. The interval of attained gradient values of WGAN-SN are not included as they are always zero. It can be observed that the shrinking interval of attained gradient values avoids the saturating and exploding parts of the loss function. For example when $k_{SN} = 5.0$, the gradient of the NS-GAN loss function saturates to a value around 0 while that of the LS-GAN loss function explodes to 24.041. However, such problems do not happen when $k_{SN} \leq 1.0$. Note that we only compute the interval of attained gradient values on one of the two symmetric loss terms used in the discriminator

Table 3: FID scores vs. k_{SN} (typically fixed as 1 [21]) on different datasets. When $k_{SN} \leq 1.0$, all GANs have similar performance except the WGANs (slightly worse). For the line plots, x-axis shows k_{SN} (in log scale) and y-axis shows the FID scores. From left to right, the seven points on each line have $k_{SN} = 0.2$, 0.25, 0.5, 1.0, 5.0, 10.0, 50.0 respectively. Lower FID scores are better.

Dataset	GANs	FID Scores						
		$k_{SN} = 0.2$	0.25	0.5	1.0	5.0	10.0	50.0
MNIST	NS-GAN-SN	5.41	3.99	4.20	3.90	144.28	156.60	155.41
	LS-GAN-SN	5.14	**3.96**	**3.90**	4.42	36.26	59.04	309.35
	WGAN-SN	6.35	6.06	4.44	4.70	**3.58**	**3.50**	**3.71**
	COS-GAN-SN	5.41	4.83	4.05	3.86	291.44	426.62	287.23
	EXP-GAN-SN	**4.38**	4.93	4.25	**3.69**	286.96	286.96	286.96
CIFAR10	NS-GAN-SN	29.23	**24.37**	23.29	**15.81**	41.04	49.67	48.03
	LS-GAN-SN	**28.04**	26.85	23.14	17.30	33.53	39.90	349.35
	WGAN-SN	29.20	25.07	26.61	21.75	**21.63**	**21.45**	**23.36**
	COS-GAN-SN	29.45	25.31	**20.73**	15.88	309.96	327.20	370.13
	EXP-GAN-SN	30.89	24.74	20.90	16.66	401.24	401.24	401.24
CelebA	NS-GAN-SN	18.59	12.71	**8.04**	6.11	18.95	17.04	184.06
	LS-GAN-SN	20.34	**12.14**	8.85	5.69	12.40	13.14	399.39
	WGAN-SN	23.26	17.93	8.48	9.41	**9.03**	**7.37**	**7.82**
	COS-GAN-SN	20.59	13.93	8.88	**5.20**	356.70	265.53	256.44
	EXP-GAN-SN	**18.23**	13.65	9.18	5.88	328.94	328.94	328.94



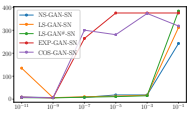
(a) MNIST (b) CIFAR10 (c) CelebA

loss function (Table 1). The interval of attained gradient values of the other loss term follows similar patterns.

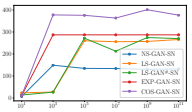
FID scores vs. Lipschitz Regularization. Table 3 shows the FID scores of different GAN loss functions with different k_{SN}. It can be observed that:

– When $k_{SN} \leq 1.0$, all the loss functions (including the non-standard ones) can be used to train GANs stably. However, the FID scores of all loss functions slightly worsen as k_{SN} decreases. We believe that the reason for such performance degradation comes from the trick used by modern optimizers to avoid divisions by zero. For example in RMSProp [26], the moving average of the squared gradients are kept for each weight. In order to stabilize training, gradients are divided by the square roots of their moving averages in each

Table 4: FID scores vs. α. For the line plots, the x-axis shows α (in log scale) and the y-axis shows the FID scores. Results on other datasets are shown in the supplementary material. Lower FID scores are better.

Dataset	GANs	FID Scores $\alpha = 1e^{-11}$	$1e^{-9}$	$1e^{-7}$	$1e^{-5}$	$1e^{-3}$	$1e^{-1}$	Line Plot
CelebA	NS-GAN-SN	9.08	7.05	7.84	18.51	18.41	242.64	
	LS-GAN-SN	135.17	6.57	10.67	13.39	17.42	311.93	
	LS-GAN$^{\#}$-SN	6.66	5.68	8.72	11.13	14.90	383.61	
	COS-GAN-SN	8.00	6.31	300.55	280.84	373.31	318.53	
	EXP-GAN-SN	8.85	6.09	264.49	375.32	375.32	375.32	

(a) $k_{SN} = 50.0$

Dataset	GANs	FID Scores $\alpha = 1e^{1}$	$1e^{3}$	$1e^{5}$	$1e^{7}$	$1e^{9}$	$1e^{11}$	Line Plot
MNIST	NS-GAN-SN	6.55	148.97	134.44	133.82	130.21	131.87	
	LS-GAN-SN	23.37	26.96	260.05	255.73	256.96	265.76	
	LS-GAN$^{\#}$-SN	13.43	26.51	271.85	212.74	274.63	269.96	
	COS-GAN-SN	11.79	377.62	375.72	363.45	401.12	376.39	
	EXP-GAN-SN	11.02	286.96	286.96	286.96	286.96	286.96	

(b) $k_{SN} = 1.0$

update of the weights, where a small positive constant ϵ is included in the denominator to avoid dividing by zero. When k_{SN} is large, the gradients are also large and the effect of ϵ is negligible. While when k_{SN} is very small, the gradients are also small so that ϵ can significantly slow down the training and worsen the results.

- When $k_{SN} \leq 1.0$, the performance of WGAN is slightly worse than almost all the other GANs. Similar to the observation of [12], we ascribe this problem to the shifting domain of WGANs (Figure 4 (c)(f)). The reason for the domain shift is that the Wasserstein distance only depends on the difference between $\mathbb{E}[f(x)]$ and $\mathbb{E}[f(g(z)]$ (Table 1). For example, the Wasserstein distances $\mathbb{E}[f(x)] - \mathbb{E}[f(g(z)]$ are the same for i) $\mathbb{E}[f(x)] = 0.5, \mathbb{E}[f(g(z)) = -0.5$ and ii) $\mathbb{E}[f(x)] = 100.5, \mathbb{E}[f(g(z)) = 99.5$.
- When $k_{SN} \geq 5.0$, the WGAN works normally while the performance of all the other GANs worsen and even break (very high FID scores, e.g. ≥ 100). The reasons for the stable performance of WGAN are two-fold: i) due to the KR duality, the Wasserstein distance is insensitive to the Lipschitz constant K. Let $W(\mathbb{P}_r, \mathbb{P}_g)$ be the Wasserstein distance between the data distribution $\mathbb{P}_r$ and the generator distribution $\mathbb{P}_g$. As discussed in [2], applying K-Lipschitz regularization to WGAN is equivalent to estimating $K \cdot W(\mathbb{P}_r, \mathbb{P}_g)$,

which shares the same solution as 1-Lipschitz regularized WGAN. ii) To fight the exploding and vanishing gradient problems, modern neural networks are intentionally designed to be scale-insensitive to the backpropagated gradients (e.g. ReLU [6], RMSProp [26]). This largely eliminates the scaling effect caused by k_{SN}. This observation also supports our claim that the restriction of the loss function is the dominating factor of the Lipschitz regularization.
– The best performance is obtained by GANs with strictly convex (e.g. NS-GAN) and properly restricted (e.g. $k_{SN} = 1$) loss functions that address the shifting domain and exploding/vanishing gradient problems at the same time. However, there is no clear preference and even the non-standard ones (e.g., COS-GAN) can be the best. We believe that this is due to the subtle differences of the convexity among loss functions and propose to leave it to the fine-tuning of loss functions using the proposed domain scaling.

Qualitative results are in the supplementary material.

4.3 Empirical Results on Domain Scaling

In this section, we empirically verify our claim that the restriction of the loss function is the dominating factor of the Lipschitz regularization. To illustrate it, we decouple the restriction of the domain of the loss function from the Lipschitz regularization by the proposed domain scaling method (Section 3.2).

Table 4 (a) shows that (i) the FID scores of different loss functions generally improve with decreasing α. When $\alpha \leq 10^{-9}$, we can successfully train GANs with extremely large Lipschtiz constant ($K \approx k_{SN}^n = 50^4 = 6.25 \times 10^6$), whose FID scores are comparable to the best ones in Table 3. (ii) The FID scores when $\alpha \leq 10^{-11}$ are slightly worse than those when $\alpha \leq 10^{-9}$. The reason for this phenomenon is that restricting the domain of the loss function converges towards the performance of WGAN, which is slightly worse than the others due to its shifting domain. To further illustrate this point, we scale the domain by $\alpha = 1e^{-25}$ and show the FID scores of WGAN-SN and those of different loss functions in Table 5. It can be observed that all loss functions have similar performance. Since domain scaling does not restrict the neural network gradients, it does not suffer from the above-mentioned numerical problem of division by zero ($k_{SN} \leq 1.0$, Table 3). Thus, it is a better alternative to tuning k_{SN}.

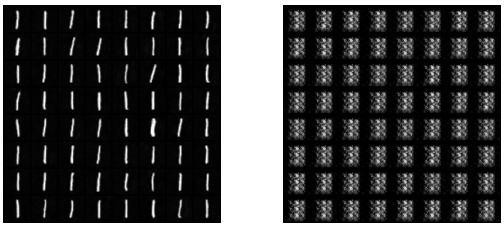
Table 5: FID scores of WGAN-SN and some extremely degenerate loss functions ($\alpha = 1e^{-25}$) on different datasets. We use $k_{SN} = 50$ for all our experiments.

GANs	FID Scores		
	MNIST	CIFAR10	CELEBA
WGAN-SN	3.71	23.36	7.82
NS-GAN-SN	3.74	21.92	8.10
LS-GAN$^{\#}$-SN	3.81	21.47	8.51
COS-GAN-SN	3.96	23.65	8.30
EXP-GAN-SN	3.86	21.91	8.22

Table 4 (b) shows that the FID scores of different loss functions generally worsen with less restricted domains. Note that when $\alpha \geq 10^5$, the common

practice of 1-Lipschitz regularization fails to stabilize the GAN training. Note that the LS-GAN-SN has some abnormal behavior (e.g. $\alpha = 1e^{-11}$ in Table 4 (a) and $\alpha = 1e^{1}$ in Table 4 (b)) due to the conflict between its 0.5-centered domain and our zero-centered domain scaling method (Eq.8). This can be easily fixed by using the zero-centered LS-GAN$^{\#}$-SN (see Table 1).

Bias over Input Samples. When weak Lipschitz regularization (large Lipschitz constant K) is applied, we observed mode collapse for NS-GAN and crashed training for LS-GAN, EXP-GAN and COS-GAN (Figure 5, more results in supplementary material). We conjecture that this phenomenon is rooted in the inherent bias of neural networks over input samples: neural networks may "prefer" some input (class of) samples over the others by outputting higher/lower values, even though all of them are real samples from the training dataset. When the above-mentioned loss functions are used, such different outputs result in different backpropagated gradients $\nabla L = \partial D(f(x))/\partial f(x)$. The use of weak Lipschitz regularization further enhances the degree of unbalance among backpropagated gradients and causes mode collapse or crashed training. Note that mode collapse happens when ∇L is bounded (e.g. NS-GAN) and crashed training happens when ∇L is unbounded (e.g. LS-GAN, EXP-GAN) or "random" (e.g. COS-GAN). However, when strong Lipschitz regularization is applied, all loss functions degenerate to almost linear ones and balance the backpropagated gradients, thereby improve the training.



(a) NS-GAN-SN (b) LS-GAN-SN

Fig. 5: (a) Mode collapse and (b) crashed training on MNIST, $k_{SN} = 50.0$, $\alpha = 1e^{-1}$.

5 Conclusion

In this paper, we studied the *coupling* of Lipschitz regularization and the loss function. Our key insight is that instead of keeping the neural network gradients small, the dominating factor of Lipschitz regularization is its restriction on the domain and interval of attainable gradient values of the loss function. Such restrictions stabilize GAN training by avoiding the bias of the loss function over input samples, which is a new step in understanding the exact reason for Lipschitz regularization's effectiveness. Furthermore, our finding suggests that although different loss functions can be used to train GANs successfully, they actually work in the same way because all of them degenerate to near-linear ones within the chosen small domains.

Acknowledgement This work was supported in part by the KAUST Office of Sponsored Research (OSR) under Award No. OSR-CRG2018-3730.

References

1. Arjovsky, M., Bottou, L.: Towards principled methods for training generative adversarial networks. In: International Conference on Learning Representations (2017)
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 214–223. PMLR, International Convention Centre, Sydney, Australia (06–11 Aug 2017), http://proceedings.mlr.press/v70/arjovsky17a.html
3. Barratt, S., Sharma, R.: A note on the inception score. arXiv preprint arXiv:1801.01973 (2018)
4. Fedus*, W., Rosca*, M., Lakshminarayanan, B., Dai, A.M., Mohamed, S., Goodfellow, I.: Many paths to equilibrium: GANs do not need to decrease a divergence at every step. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=ByQpn1ZA-
5. Frühstück, A., Alhashim, I., Wonka, P.: Tilegan: Synthesis of large-scale non-homogeneous textures. ACM Trans. Graph. **38**(4) (Jul 2019). https://doi.org/10.1145/3306346.3322993
6. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: Gordon, G., Dunson, D., Dudík, M. (eds.) Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 15, pp. 315–323. PMLR, Fort Lauderdale, FL, USA (11–13 Apr 2011), http://proceedings.mlr.press/v15/glorot11a.html
7. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 27, pp. 2672–2680. Curran Associates, Inc. (2014), http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf
8. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 5767–5777. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/7159-improved-training-of-wasserstein-gans.pdf
9. Gulrajani, I., Raffel, C., Metz, L.: Towards GAN benchmarks which require generalization. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=HkxKH2AcFm
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016)
11. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 6626–6637. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/7240-gans-trained-by-a-two-time-scale-update-rule-converge-to-a-local-nash-equilibrium.pdf
12. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=Hk99zCeAb

13. Kelly, T., Guerrero, P., Steed, A., Wonka, P., Mitra, N.J.: Frankengan: Guided detail synthesis for building mass models using style-synchonized gans. ACM Trans. Graph. **37**(6) (Dec 2018). https://doi.org/10.1145/3272127.3275065, https://doi.org/10.1145/3272127.3275065
14. Kurach, K., Lucic, M., Zhai, X., Michalski, M., Gelly, S.: The GAN landscape: Losses, architectures, regularization, and normalization (2019), https://openreview.net/forum?id=rkGG6s0qKQ
15. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (December 2015)
16. Lucic, M., Kurach, K., Michalski, M., Gelly, S., Bousquet, O.: Are gans created equal? a large-scale study. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 31, pp. 700–709. Curran Associates, Inc. (2018), http://papers.nips.cc/paper/7350-are-gans-created-equal-a-large-scale-study.pdf
17. Ma, L., Jia, X., Sun, Q., Schiele, B., Tuytelaars, T., Van Gool, L.: Pose guided person image generation. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 406–416. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/6644-pose-guided-person-image-generation.pdf
18. Mao, X., Li, Q., Xie, H., Lau, R.Y.K., Wang, Z., Smolley, S.P.: On the effectiveness of least squares generative adversarial networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**(12), 2947–2960 (2019)
19. Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S.: Least squares generative adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
20. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for GANs do actually converge? In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 3481–3490. PMLR, Stockholmsmässan, Stockholm Sweden (10–15 Jul 2018), http://proceedings.mlr.press/v80/mescheder18a.html
21. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=B1QRgziT-
22. Nowozin, S., Cseke, B., Tomioka, R.: f-gan: Training generative neural samplers using variational divergence minimization. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29, pp. 271–279. Curran Associates, Inc. (2016), http://papers.nips.cc/paper/6066-f-gan-training-generative-neural-samplers-using-variational-divergence-minimization.pdf
23. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
24. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
25. Sanyal, A., Torr, P.H., Dokania, P.K.: Stable rank normalization for improved generalization in neural networks and gans. In: International Conference on Learning Representations (2020), https://openreview.net/forum?id=H1enKkrFDB

26. Tieleman, T., Hinton, G.: Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning **4**(2), 26–31 (2012)
27. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
28. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29, pp. 82–90. Curran Associates, Inc. (2016), http://papers.nips.cc/paper/6096-learning-a-probabilistic-latent-space-of-object-shapes-via-3d-generative-adversarial-modeling.pdf
29. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Oct 2017)